Computer Science • 20(1) 2019 https://doi.org/10.7494/csci.2019.20.1.3023

Michał Jungiewicz
Aleksander Smywiński-Pohl

TOWARDS TEXTUAL DATA AUGMENTATION FOR NEURAL NETWORKS: SYNONYMS AND MAXIMUM LOSS

Abstract *Data augmentation is one of the ways to deal with labeled data scarcity and overfitting. Both of these problems are crucial for modern deep-learning algorithms, which require massive amounts of data. The problem is better explored in the context of image analysis than for text; this work is a step forward to help close this gap. We propose a method for augmenting textual data when training convolutional neural networks for sentence classification. The augmentation is based on the substitution of words using a thesaurus as well as Princeton University's WordNet. Our method improves upon the baseline in most of the cases. In terms of accuracy, the best of the variants is 1.2% (pp.) better than the baseline.*

Keywords deep learning, data augmentation, neural networks, natural language processing, sentence classification

Citation Computer Science 20(1) 2019: 57–83

1. Introduction

This work approaches the notion of data augmentation (DA) in training neural networks for solving the text classification problem. DA is based on the idea that the automatic modification of the data used to train a neural network (or another classification algorithm) can improve its performance. The modifications are designed in a way that should preserve the label assigned to the original piece of data. For example – the small rotation of an image of a triangle results in a new data sample but keeps the original (triangle) label intact. Such a procedure can be used in cases when the data acquisition process is particularly costly as well as for producing more diversified samples, making the network less prone to overfitting.

The so-called deep learning has improved the state of the art in multiple different fields [23]. It gives stunning results and is definitely the focus not only of academia but also of the industry. Most of the work has been done with visual data, but a deep-learning tsunami (as Christopher Manning puts it) is also engulfing the field of natural language processing (NLP) [26].

Although it provides very promising results, deep learning also has its shortcomings. Among others, there is a problem of labeled data scarcity. This kind of data is most valuable for deep-learning algorithms, as generally supervised models provide more accurate results than unsupervised models. The data-labeling process is time-consuming and expensive. Moreover, the performance of a single data annotator may deteriorate with time, since many of the tasks are extremely repetitive and boring. As a result, the annotators who are paid for each piece of data often optimize their efforts and are less focused on the task. For example, a comparative study of cyberbullying annotation by means of Mechanical Turk and a group of carefully selected annotators [34] showed that the latter group performed much better, since all of the tested cyberbullying detection systems, as well as the machine-learning algorithm trained on the latter annotation, gave higher scores on the data set annotated by the second group. The data annotation may also face legal issues, especially in cases when personal data is processed.

Another problem that is also difficult to tackle is overfitting, meaning a large difference between the performance of a given data-trained model on the training and the test sets. If the model has a large-enough number of parameters, the model memorizes all of the training examples instead of generalizing the patterns present in the data; as a result, it shows poor performance in real-world scenarios. The primary means of tackling this challenge is using dropout [42] – "turning off" large parts of the network during the training process, which forces it to generalize rather than memorize the patterns. But DA can also be used, as the observed data is different in each epoch. To sum up, DA can reduce data annotation costs and overfitting.

This work focuses specifically on textual data augmentation. As with deep learning in general, more work has been done for visual data than for textual data. This paper helps to close this gap. What is more, DA enables us to leverage existing language resources (e.g., dictionaries) and integrate them into machine-learning methods.

Finally, there is a need for a more systematic approach to DA. Very often, techniques of augmenting data are more of an improvisation than an evidence-based procedure. This paper is a step towards a systematic approach to the matter.

Our solution builds upon the method proposed in [13] for augmenting visual data. The authors of this paper present a technique for choosing augmented data samples. By maximizing the current loss of a classifier on the new data sample, an overfitting reduction can be achieved. We transferred this approach to the language domain, with synonym replacement using available language resources as a primary text transformation. We tested it on a sentence-classification problem and made use of the convolutional network models proposed by Yoon Kim [17] and Sosuke Kobayashi [19].

Our results are promising because models leveraging our DA technique performed better than the baseline in most cases. On the other hand, this did not happen in all cases; this is also a very interesting finding. The best of our variants was 1.2% (percentage points) better than the baseline in terms of accuracy. We also conclude that a dictionary extracted from Thesaurus.com was generally a better resource for augmenting data in our case than Princeton's WordNet [30]. This is also an important conclusion; it shows that not all resources are equal in terms of boosting the effectiveness of neural models using DA.

2. Related work

One of the most notable trends now in natural language processing is the application of deep learning for its tasks, whereas data augmentation is one of the investigated ways for improving its achievements.

2.1. Deep learning for NLP

Deep learning for NLP has shown its achievements in multiple different areas. For example, there has been a surge of interest in pre-trained language models used for different NLP tasks [50]. One of the most promising models was proposed by Devlin et al. [9]; the authors called it BERT – Bidirectional Encoder Representations from Transformers. They introduced a new language representation model that improves on the state-of-the-art results in multiple NLP tasks. The distinguishing feature of this model is that it conditions on both the left and right context in all layers.

Another NLP task at which deep learning has shown to excel is machine translation. Researchers have proposed multiple different approaches for neural machine translation systems. Some of the most effective models include the attention mechanism [45], CNN-based seq2seq learning [14], and deep LSTM network with residual and attention connections [49].

A further area of application for deep learning has been dialogue systems. Here, some of the best models are comprised of one employing the LSTM encoder on top

of sentence-level CNN embeddings [53], another model leveraging a memory network [11], and a different method using LSTMs to encode the message and response with the inner product for ranking [25].

2.2. Overview of data augmentation

Data augmentation is not intrinsically connected to neural networks; however, this work focuses specifically on using DA as a part of training neural networks.

Deep learning has become one of the most popular topics in recent computer science research. Most of the work has been done with visual data [5, 6, 13, 32] yet less with textual data, although more and more papers have been published on deep learning for NLP [7, 41, 52] (see Figure 1). The same phenomenon occurs in data augmentation for deep learning. There is considerably more work that treats the visual data augmentation topic than that of textual data. There is also research that focuses on more general methods for DA, which can be used in the context of different data types: images, texts, etc.

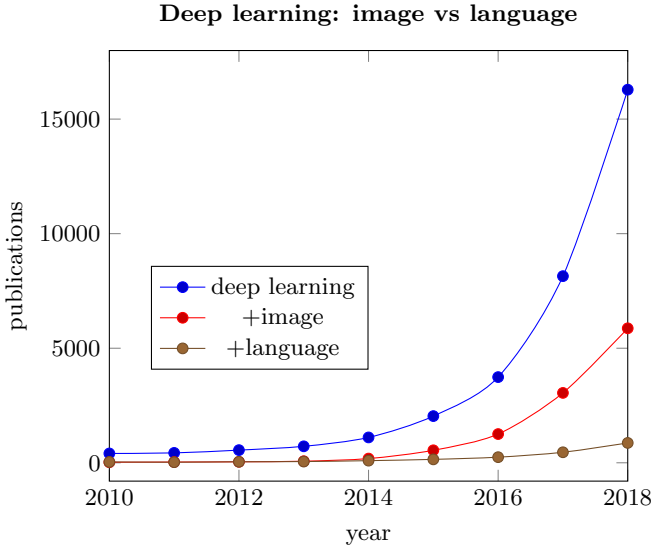

Figure 1. Deep-learning trends according to app.dimensions.ai website [2]. We searched for keywords in abstracts and titles. We looked only for phrase "deep learning" and then "deep learning" plus "image" and "deep learning" plus "language"

2.3. General data augmentation

One approach for general data augmentation is to replicate the data in feature-space instead of data-space. The authors of [47] propose one method for augmenting data in data-space and another for feature-space. They compare the results and find out

that, if we know the plausible kinds of transformations on the data, the data-space approach surpasses the feature-space approach.

Another way of dealing with the general case is proposed by the authors of [37]. Given a user-specified set of data transformations, they train a generative sequence model that can then be used to augment both visual and textual data. Their method is unsupervised, so it uses unlabeled data for training.

2.4. Image data augmentation

Within the visual data augmentation problem, multiple approaches can be observed. Presumably, the most obvious solution is to replicate the data randomly with carefully selected types of transformations [5, 6]. The authors of [5] use affine transformations on images (translation, rotation, scaling, horizontal shearing) and elastic deformations taken from [40]. The authors of [6] use similar augmentation techniques: rotations, translations, and scaling (among others).

A greedy algorithm to solve the issue is proposed by Paulin et al. [32], which offers promising results but is computationally expensive. The authors introduce the Image Transformation Pursuit (ITP) algorithm to choose the right types of transformations; however, this approach is time-consuming because it involves multiple cases of classifier retraining.

Finally, the authors of [13] propose a method that does not involve classifier retraining. On the other hand, they choose examples for DA by maximizing the current loss of the classifier. This approach should intuitively reduce overfitting and, thus, increase the overall effectiveness of the trained model. This work builds upon their work, using the loss maximization approach and transferring this idea to the language domain.

2.5. Textual data augmentation

One work that approaches the problem of textual data augmentation is [52]. It is not the main focus of the work, but DA is used as a way to improve the training performance of a convolutional neural network. The authors use a thesaurus for the augmentation.

Ryan R. Rosario proposes a preprocessing approach that uses DA [38]. The author classifies short texts, whereas this work focuses on sentence classification. The other difference is the employed classifier: Rosario uses SVM, while this work concentrates on neural networks. Moreover, Rosario creates longer texts out of the original fragments, while we focus on transforming sentences without changing their lengths.

Jonathan Quijas has published research on DA for texts [36]. Among other issues, the author examines the effect of DA techniques on training convolutional and recurrent neural networks for text classification. However, the work is different from ours because it takes a different approach. The author examines shuffling, noise injection, and padding. Shuffling means randomly changing the order of words within

a small context window. Noise injection is changing a word to a different randomly chosen one. Padding means filling the input with the words from a sentence instead of null characters. Our approach, on the other hand, examines synonym replacement and leverages the maximum loss approach from [13]. What is more, the samples for augmentation are chosen during training, not as preprocessing. The rationale for why this difference is important is given in Section 4.3.

More recently, Sosuke Kobayashi proposed another approach [19]. The author suggests replacing the word in a sentence by its counterparts generated with a bi--directional language model. Sosuke calls the method "contextual augmentation". The author shows improvements across multiple datasets. To our knowledge, this is the only paper that proposes textual data augmentation during training, which is also our approach. On the other hand, Sosuke proposes a different method for transforming the data.

Claude Coulombe has also published a paper on textual data augmentation for neural networks [8]. The author tested multiple different approaches: textual noise, spelling errors, synonyms replacement, paraphrase generation (using regular expressions or syntax trees), and back-translation. Claude performs tests on different network architectures; the difference between this work and ours is that it augments the data as a preprocessing stage, not during training.

3. Background

We introduce the essential theoretical background for understanding the problem we are trying to solve. It depicts some of the machine-learning concepts and methods we leverage as well as the language resources and dataset we use.

3.1. Classification

Classification is one of the typical problems that machine learning is trying to address [15]. The problem is the assignment of one of k predefined categories to a given previously unobserved input object. The learning algorithm that involves a neural network should typically produce a function ($f : \mathbb{R}^n \rightarrow \{1, ..., k\}$) that maps the input of real values to a single category. If the task is to assign a single category, it is called single-label classification, but there is also multi-label classification, where multiple categories can be assigned to a single data sample.

3.2. TREC dataset

This paper focuses on the single-label classification of English sentences. For testing our model, we used the TREC dataset [24], where questions have to be classified into predefined sets of types. Table 1 summarizes the primary statistics of the dataset. The data is split into six disjoint classes: ABBR (abbreviation), DESC (description and abstract concepts), ENTY (entities), HUM (human beings), LOC (locations), and NUM (numeric values). Actually, the TREC dataset is split into more specific

categories within these six general classes; however, in the paper by Kim [17] as well as in our work (which builds on top of it), only these general categories are taken into account. Table 2 shows example sentences for each of these six categories.

Table 1
Statistics for TREC dataset as given by [17]

Classes	6
Average sentence length	10
Dataset size	5952
Vocabulary size	9592
Number of words present in word2vec	9125
Test size	500

Table 2
Examples of sentences from TREC dataset. One example for each class

ABBR What is the full form of .com?
DESC How can I find a list of celebrities' real names?
ENTY What films featured the character Popeye Doyle?
HUM Who killed Gandhi?
LOC What sprawling U.S. state boasts the most airports?
NUM What is the date of Boxing Day?

3.3. Neural networks

Artificial neural networks (ANNs) are models that share some common characteristics with biological neural networks [12]. ANNs consist of simple elements called neurons that are connected to each other by links. Each link has a weight that multiplies the signal that is passed over it. The neuron applies an activation function to the sum of its inputs and outputs the output of the function. The following formula describes it: $y = f(w_1x_1 + w_2x_2 + ... + w_nx_n)$, where y is the output of a neuron, f is an activation function, w_n is the weight of the nth input to the neuron, and x_n is the value of the nth input to the neuron.

3.4. Backpropagation algorithm

The most basic and widely used method for training ANNs is the backpropagation algorithm [22,31,39,46]. As the author of [12] puts it, the backpropagation algorithm

is simply a gradient descent method for minimizing the total squared error of the output of an ANN.

The training of an ANN by backpropagation is comprised of four stages:

- *propagation* of the signal forwards through the net (from the input to the output layer),
- *calculation* of the error (by comparing the actual and expected value at the output layer),
- *backpropagation* of this error (from the output layer to the input layer),
- *adjustment* of weights according to the computed error and learning rate.

3.5. Convolutional neural networks

A convolutional neural network (CNN) is a suitable model for processing data that has a well-known grid-like topology [15]. CNNs are used predominantly in image processing, but they can also be found in NLP and other fields. Their distinctive feature is the convolution operation that is used instead of general matrix multiplication, as in fully connected feedforward ANNs.

In the continuous domain, the convolution can be described by the following formula:

$$s(t) = \int x(a)w(t-a)da \tag{1}$$

where x and w are continuous functions, x is the input function, and w the kernel function. The name kernel and filter are often used interchangeably in this context. The output of convolution is sometimes called a feature map. The operation is often denoted with an asterisk:

$$s(t) = (x * w)(t) \tag{2}$$

In the discrete version, it can be defined as follows:

$$s(t) = \sum_{-\infty}^{\infty} x(a)w(t-a) \tag{3}$$

Convolutional networks are organized in layers. For grayscale image processing, each layer of convolution has two dimensions – width and height. To process an image with colors, you need another dimension (which is most often called channels). Each channel corresponds to one of the RGB colors.

Convolutional networks have multiple features that facilitate data processing. They have sparse connectivity, which is accomplished by making the kernel considerably smaller than the input. Typically, kernels are of a constant size that is independent of the data size. We give some examples of kernel sizes from the literature to provide a better picture. The famous Alexnet paper [21] states the following sizes: 11×11×3 (11 by 11 pixels window spanning 3 channels) in the first layer, 5×5×48, 3×3×256, 3×3×192, and 3×3×192 in the subsequent layers. Yoon Kim's paper on sentence classification mentions filters of sizes 3, 4, and 5 [17]. The sparse connectivity

feature of CNNs results in the simplification of the model, which in turn enables better memory, computational, and statistical efficiency.

Another feature of convolutional networks is called parameter sharing. Instead of learning a separate set of parameters for each part of the data (e.g., the region of an image), one set is learned. It is one of the key features of CNNs, which distinguishes them from more-basic models such as feedforward fully connected MLPs without convolutional layers. This feature also improves memory and statistical efficiency.

A pooling operation is used as a part of the convolutional network data processing. It consists of replacing the value of a few neighboring outputs with a summary statistic. This statistic can be a maximum, minimum, average, etc. depending on the task that the network must solve. Pooling makes CNNs invariant to small translations of the data. Pooling can also be useful for handling inputs of varying size, which is needed in sentence classification, for example.

3.6. Transformer architecture

The CNN architecture has been used for a long time in research. One of the most recent neural network architectures that is specially designed for NLP tasks is the Transformer. It is proposed in Google's paper: ,,Attention is all you need" [45].

The authors of the Transformer state that it is simpler than convolutional and recurrent neural networks (RNNs). Instead of using CNNs or RNNs, it leverages only the attention mechanism that connects the encoder and decoder. It achieves impressive results, especially in machine translation (for which it is designed). What is more, it turns out that it applies well to English constituency parsing. On top of being superior in effectiveness, it is also more parallelizable and requires significantly less time to train. It has also been used for text classification.

Multiple authors have built upon the Transformer architecture. One of these works proposes the use of the model for training a character-level language model [1]. In this paper, the authors use a deep transformer model, which surpasses RNN variants on two popular benchmarks: text8 and enwik8.

3.7. Word2vec

Word2vec is a popular name for a set of models that have recently been widely used and studied [27, 28]. Word2vec is a distributed representation of words. Unlike the simple bag-of-words representation, each word is represented as a vector of continuous values that uses one hot encoding (only one value is 1 – the rest are 0). In a bag-of--words vector space, every two words are orthogonal to each other, and the vectors representing the words have as many dimensions as the size of the whole vocabulary (which reaches hundreds of thousands of words for a typical corpus). On the other hand, word2vec vectors have fewer dimensions – for example, a few hundred. Therefore, the word2vec model can (and in fact does) capture some meaning similarity between words.

Since Tomas Mikolov proposed the word2vec model, various teams from around the world have tried to come up with a different approach for obtaining word embeddings. Among others, we might mention Stanford's GloVe [33] and Facebook's Fasttext subword embeddings [3].

3.8. Synonym sources

For synonym replacement (see Section 4), we used two sources in our solution: WordNet [29, 30, 48], and Thesaurus.com [43].

Princeton's WordNet is a large lexical database for English. In this database, words are grouped in so-called synsets that gather synonyms. Words inside the synsets are synonyms, but synsets between each other are connected by multiple other kinds of relations. These include antonymy, hyponymy, and hypernymy (super-subordinate relationship or ISA relationship), meronymy (part-whole relationship), troponymy (which is to verbs what hyponymy is to nouns), and entailment. These relationships are also a part of WordNet; by using WordNet, it is possible to extract words connected by these relationships. Table 3 shows the size of WordNet, taking into account the different parts of speech [48]. These statistics apply to WordNet Version 3.0. In this paper, Princeton's WordNet is referred to as WordNet.

Table 3
Size of WordNet, taking into account different parts of speech [48]

Part of speech	Unique strings	Synsets	Total word-sense pairs
Noun	117,798	82,115	146,312
Verb	11,529	13,767	25,047
Adjective	21,479	18,156	30,002
Adverb	4481	3621	5580
Totals	155,287	117,659	206,941

Thesaurus.com is a website that is connected to Dictionary.com. The owners state on their site that Dictionary.com is the world's leading digital dictionary [43]. They provide millions of English definitions, spellings, audio pronunciations, example sentences, and word origins. Thesaurus.com and Dictionary.com were created in 1995; in 2014, they surpassed 70 million monthly users. Thesaurus.com is a service that finds synonyms for a given word. The site is used as a source of synonyms by the Python library called PyDictionary [35], which we used in the experiments shown in this paper. We refer to Thesaurus.com as Thesaurus.

4. Solution

The gist of this paper is our proposed solution for the data augmentation of texts. We used two similar CNN models for classifying sentences and modified them to use our method for transforming the sentences while training the network.

4.1. Problem statement

This paper focuses on solving the sentence classification task with the use of the convolutional neural networks model. It examines the influence of the DA technique proposed by us on the effectiveness of this model in solving the task.

4.2. Classifier

As a model for sentence classification, we chose the CNN architecture proposed by Yoon Kim [17]. It is trained on top of pre-trained word vectors – word2vec [28]. The model is used for sentence classification. The general view of the architecture of this model is shown in Figure 2. Looking from the left side, we first have a layer of word2vec vectors; then, there is a convolutional layer with multiple filter widths and feature maps. After that, there is a pooling layer; and finally, a fully connected layer with dropout and softmax output.

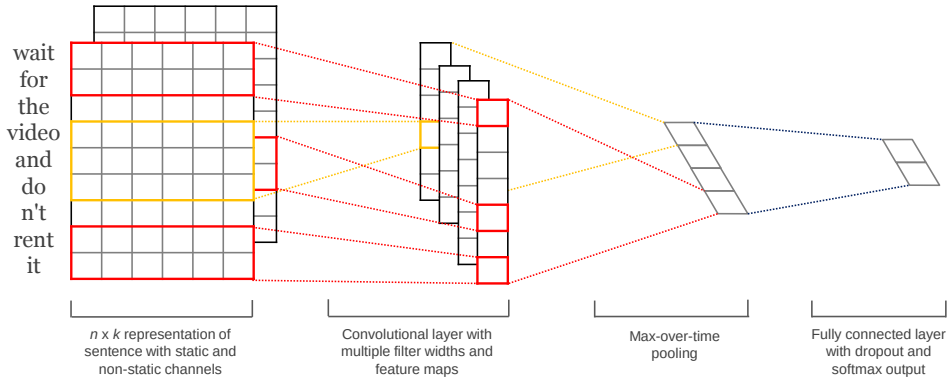

Figure 2. Illustration of Kim CNN model architecture

We chose this because it improves upon the state of the art in some NLP tasks. What is more, ANNs have recently been the focus of NLP research [50]. The Kim CNN model is not deep, as it has only one layer of convolution; however, it achieves good results, and its simplicity makes it easier to use in experiments. Moreover, the distributed representations of words used by Kim have also been heavily researched recently.

The Kim CNN model comes in different variants: CNN-rand, CNN-static, CNN--non-static, and CNN-multichannel. We used two of these four variants: static and non-static. In the static variant, the word vectors are initialized with word2vec but do not change during training, whereas they change during training in the non-static variant. The architecture shown in Figure 2 is for two channels of word2vec vectors. Both channels are used only in the CNN-multichannel variant (which we do not test in this paper). It is a hybrid of static and non-static architectures. One of the channels holds the word2vec embeddings that are changed by backpropagation during

training, and the other channel consists of vectors that keep their values throughout the training. We used the Kim CNN model for the experiments shown in Section 5.1. The second baseline model we used was implemented by Sosuke Kobayashi [19,20]. It is similar to the Kim CNN architecture, and we used it in Section 5.2. We used only the baseline Kobayashi's model – not the one Sosuke proposed and called "contextual augmentation".

4.3. Data augmentation

The main topic of this paper is to determine how data augmentation techniques can improve the effectiveness of CNNs used for classifying sentences. Therefore, it is the most important part of our solution.

We modified the Kim CNN model according to the solution proposed by Fawzi et al. [13] for augmenting visual data. The idea of the work is as follows. The possible space of transformations for augmenting data is theoretically infinite; therefore, it is a challenge to determine which types of changes should be performed. Certain augmentation transformations can be beneficial to our model, but others are not necessarily so. There should be a way to choose the best ones; i.e., those that increase the effectiveness of the model. Fawzi et al. propose choosing the transformations that maximize the current loss of the classifier. Intuitively, it should reduce overfitting, because the classifier is trained on examples that are "least familiar" to the classifier in its current state. The authors explain the intuition in their paper with a simple example. According to this idea, they modify the step of stochastic gradient descent (SGD) algorithm [4]. With probability p, they transform the example used for training. With probability $1 - p$, they leave the original sample for training.

The last question in our case is this – what types of DA transformations should we use for texts? The literature introduces the term label preserving transformations (LPTs) for visual data augmentation; this is used in [47], for example. It defines those transformations that do not change the meaning of the data.

As LPT for text, we propose synonym replacement according to the procedure described below. Actually, this procedure ensures that the transformations preserve the labels in most or all cases, but the transformed sentence might not be linguistically correct. This is because the replaced synonym might not fully fit the context that is shown in Table 4. The words in bold are replacements that are linguistically correct; the other replacements make some sense but do not fit the context. The replacements shown in Table 4 come from Thesaurus [43].

To cope with the problem of choosing synonyms that do not fit the context, we restrain the replacement of only those words that belong to certain parts of speech. This solution does not solve the problem completely; on the other hand, it is a simple way of employing synonyms as LPTs. What is more, it is not certain what is the effect of those replacements that do not fit the context but have a similar meaning. They might as well boost the ultimate effectiveness of the model.

Table 4

Example of sentence and its transformations done by synonym replacement. Changes linguistically correct are shown in bold. Replacements in this example come from Thesaurus [43]

Original sentence
aspartame is known by what other name?
Transformations
aspartame is known by what new name?
aspartame is known by what alternative name?
aspartame is known by what more name?
aspartame is known by what another name?
aspartame is known by what auxiliary name?

Our exact solution to the problem of sentence classification using DA is as follows. For each mini-batch in SGD, we use our augmentation procedure (Algorithm 1) with probability p. For part-of-speech tagging, we use the Stanford part-of-speech tagger [44]. The Kim CNN model is used for sentence classification, and so is our model. We do not try multiple replacements but only one replacement of a word per sentence.

Algorithm 1 Transformation of sentences in mini-batch

$POS(W)$ returns part of speech tag for word W

$SYNONYMS(W)$ returns W's synonyms

$REPLACE(x, y, z)$ replaces y with z in x

POS_tags is a set of allowed part of speech tags

$LOSS(M, S)$ returns loss for sentence S on model M in its current state

Input: A mini-batch MB that contains multiple sentences
Output: Transformed mini-batch

for each sentence $S_i \in MB$ **do**
  $max\_loss \leftarrow 0$
  $best\_sentence \leftarrow S_i$
  for each word $W_j \in S_i$ **do**
    $POS\_tag \leftarrow POS(W_j)$
    if $POS\_tag \in POS\_tags$ **then**
      for each $SYN_k \in SYNONYMS(W_j)$ **do**
        $S_i^{TR} \leftarrow REPLACE(S_i, W_j, SYN_k)$
        $loss \leftarrow LOSS(M, S_i^{TR})$
        if $loss > max\_loss$ **then**
          $max\_loss \leftarrow loss$
          $best\_sentence \leftarrow S_i^{TR}$
        end if
      end for
    end if
  end for
  $REPLACE(MB, S_i, S_i^{TR})$
end for

The novelty of our solution lies in the following achievements. First, it transfers the technique of DA proposed by [13] from images to text. Second, it tests the basic data transformations for text (namely, synonym substitution) using two different resources. Our proposed solution improves upon the baseline Kim CNN model in most cases (as is shown in Section 5). Finally, our work is a step towards a systematic approach to DA for textual data in the context of ANN models.

5. Results and discussion

We performed two sets of experiments. The first one (shown in Section 5.1) checked the effectiveness of our DA method for different values of its hyperparameters. We used the Kim CNN model here. In the second set of tests, we used Kobayashi's implementation (see Section 5.2). We checked if the DA variant that performed the best on the validation set was better than the baseline there. We also added some statistical analysis at the end of the latter part.

5.1. Tests for different parameters

For this set of tests, we used the Harvard implementation of the Kim CNN model [16] with its default parameters (shown in Table 5). For each test, we averaged the results for five different seeds of a random number generator. We tested our model on the TREC dataset [24] with the specific task of determining the types of subjects in the user questions. The dataset consists of around 6000 questions labeled with the 6 subject-type categories given in Table 2.

Table 5
Default parameters of Harvard Kim CNN implementation

Size of mini-batch	50
Optimization method	ADADELTA [51]
Dropout [42] probability	0.5
L2 norm of final linear layer weights	3
Number of epochs to train	25
Kernel sizes of different convolutions	3, 4, 5
Number of convolution feature maps	100

Our solution has the following hyper-parameters:

- *presence* or *absence* of dropout;
- probability of choosing the transformed mini-batch for training instead of original mini-batch p: 0.25, 0.5, 0.75;
- part of speech that is replaced: *noun*, *adjective* or *both*;
- source of synonyms that is used for replacements: *WordNet* or *Thesaurus*.

The results for the baseline Harvard Kim CNN implementation with default parameters and without our DA method are shown in Table 6.

Table 6

Results for baseline Harvard Kim CNN implementation on TREC dataset. Results present accuracy (AC); i.e., proportion of total number of predictions that were correct

	With dropout [%]	Without dropout [%]
Static	92.32	92.64
Non-static	92.44	92.72

The results of our own solution are shown in both Tables (7 to 9) and on plots to easier capture the relative differences between the approaches (Figures 3 to 5). The tables show the results for the different parts of speech of the words that are replaced: Table 7 is for nouns, 8 for adjectives, and 9 for both. The plots are organized in the same way: Figure 3 is for nouns, 4 for adjectives, and 5 for both. In all cases, the results show the accuracy on the test set for the model that gave the best result on the validation set throughout training. We did not perform test time augmentation, so we did not change the test dataset from the original.

Table 7

Results of classification on TREC dataset. Only nouns are replaced. Kim CNN static or non-static model with or without dropout. Scores for different values of p, which is probability of choosing transformed mini-batch for training instead of original mini-batch. Results present accuracy (AC); i.e., proportion of total number of predictions that were correct. For each set of variants, best result is shown in bold

Nouns, WordNet				
Model	$p =$	0.25	0.5	0.75
dropout	static	92.16%	92.36%	92%
	non-static	92.12%	92%	91.8%
no dropout	static	92.48%	92.76%	92.16%
	non-static	92.32%	92.04%	91.88%
(Augmented with $p = 0.5$, no dropout, static) = 92.76%				
Nouns, Thesaurus				
Model	$p =$	0.25	0.5	0.75
dropout	static	92.16%	92.04%	92.28%
	non-static	93.16%	93.16%	93.2%
no dropout	static	92.72%	93.2%	92.08%
	non-static	92.68%	92.68%	92.08%
(Augmented with $p = 0.75$, dropout, non-static) = 93.2%				
Nouns, WordNet + Thesaurus				
Model	$p =$	0.25	0.5	0.75
dropout	static	92.96%	92.96%	93.36%
	non-static	93%	93.28%	93%
no dropout	static	92.92%	93.2%	93.2%
	non-static	93%	92.88%	93.12%
(Augmented with $p = 0.75$, dropout, static) = 93.36%				

Table 8

Results of classification on TREC dataset. Only adjectives are replaced. Kim CNN static or non-static model with or without dropout. Scores for different values of p, which is probability of choosing transformed mini-batch for training instead of original mini-batch. Results present accuracy (AC); i.e., proportion of total number of predictions that were correct. For each set of variants, best result is shown in bold

Adjectives, WordNet				
Model	$p =$	0.25	0.5	0.75
dropout	static	93%	93%	92.96%
	non-static	93.12%	92.84%	93.08%
no dropout	static	93.48%	92.92%	92.8%
	non-static	93.2%	93.24%	93.12%
(Augmented with $p = 0.25$, no dropout, static) = 93.48%				
Adjectives, Thesaurus				
Model	$p =$	0.25	0.5	0.75
dropout	static	92.32%	92.8%	92.64%
	non-static	93.04%	92.96%	93.32%
no dropout	static	92.76%	92.96%	93.28%
	non-static	93.2%	92.56%	93.56%
(Augmented with $p = 0.75$, no dropout, non-static) = 93.56%				
Adjectives, WordNet + Thesaurus				
Model	$p =$	0.25	0.5	0.75
dropout	static	93%	92.32%	92.28%
	non-static	92.88%	92.76%	91.92%
no dropout	static	92.2%	92.4%	91.04%
	non-static	92.72%	92.16%	92.28%
(Augmented with $p = 0.25$, dropout, static) = 93%				

Table 9

Results of classification on TREC dataset. Both nouns and adjectives are replaced. Kim CNN static or non-static model with or without dropout. Scores for different values of p, which is probability of choosing transformed mini-batch for training instead of the original mini-batch. Results present accuracy (AC); i.e., proportion of total number of predictions that were correct. For each set of variants, best result is shown in bold

Nouns + adjectives, WordNet				
Model	$p =$	0.25	0.5	0.75
dropout	static	92.44%	92.96%	92.48%
	non-static	92.68%	92.24%	92.4%
no dropout	static	92.68%	92.92%	92.44%
	non-static	92.52%	93.2%	92.24%
(Augmented with $p = 0.5$, no dropout, non-static) = 93.2%				
Nouns + adjectives, Thesaurus				
Model	$p =$	0.25	0.5	0.75
dropout	static	92.24%	91.88%	93.4%
	non-static	93.08%	93.12%	92.32%
no dropout	static	92.96%	92.44%	92.4%
	non-static	93.92%	93.04%	93%
(Augmented with $p = 0.25$, no dropout, non-static) = 93.92%				

Table 9 (cont.)

Nouns + adjectives, WordNet + Thesaurus				
Model	$p =$	0.25	0.5	0.75
dropout	static	92.28%	92.24%	91.72%
	non-static	93.12%	92.48%	92.76%
no dropout	static	93%	92.4%	92.72%
	non-static	93.28%	92.36%	92.16%
(Augmented with $p = 0.25$, no dropout, non-static) = 93.28%				

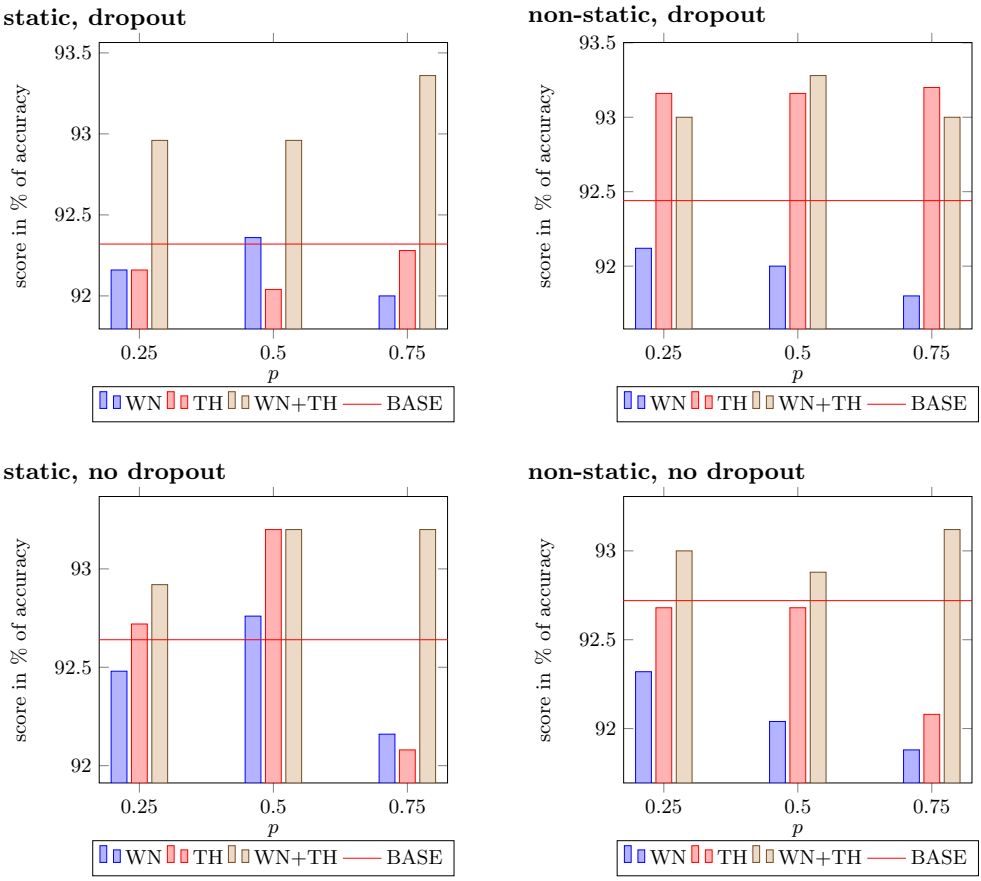

Figure 3. Results of classification on TREC dataset. Only nouns are replaced. Kim CNN static or non-static model with or without dropout. Scores for different values of p, which is probability of choosing transformed mini-batch for training instead of original mini-batch. WN means replacements are from WordNet, TH from Thesaurus, WN+TH – both. BASE is baseline result. Results present accuracy (AC); i.e., proportion of total number of predictions that were correct

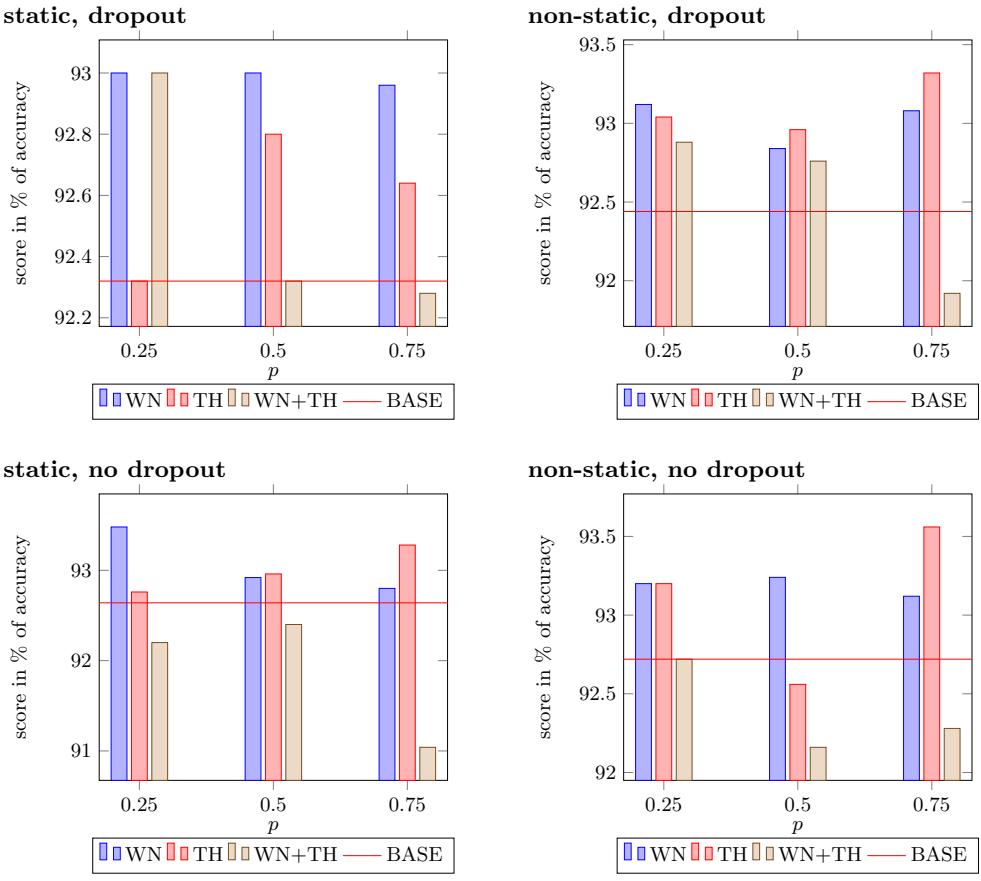

Figure 4. Results of classification on TREC dataset. Only adjectives are replaced. Kim CNN static or non-static model with or without dropout. Scores for different values of p, which is probability of choosing transformed mini-batch for training instead of the original mini-batch. WN means replacements are from WordNet, TH from Thesaurus, WN+TH – both. BASE is baseline result. Results present accuracy (AC); i.e., proportion of total number of predictions that were correct

The most important findings should answer the main research question: **does the proposed data augmentation method improve the effectiveness of the Kim CNN model on the task of sentence classification?**

The first finding is that the results of the augmentation are generally better than the baseline due to the following:

- The best overall result is **(nouns + adjectives, augmented with $p = 0.25$, no dropout, non-static, Thesaurus)**. This means that the best result uses DA. The result for this variant is 93.92%, which is 1.2% (percentage points) better as compared to the no dropout non-static baseline (which is 92.72%).

- For each set of variants, the best result is with data augmentation ($p > 0$):
  - for **(nouns, WordNet)** the best result is for **(augmented with $p = 0.5$, no dropout, static)**, which is 92.76%;
  - for **(nouns, Thesaurus)** the best result is for **(augmented with $p = 0.75$, dropout, non-static)**, which is 93.2%;
  - for **(nouns, WordNet + Thesaurus)** the best result is for **(augmented with $p = 0.75$, dropout, static)**, which is 93.36%;
  - for **(adjectives, WordNet)** the best result is for **(augmented with $p = 0.25$, no dropout, static)**, which is 93.48%;
  - for **(adjectives, Thesaurus)** the best result is for **(augmented with $p = 0.75$, no dropout, non-static)**, which is 93.56%;
  - for **(adjectives, WordNet + Thesaurus)** the best result is for **(augmented with $p = 0.25$, dropout, static)**, which is 93%;
  - for **(nouns + adjectives, WordNet)** the best result is for **(augmented with $p = 0.5$, no dropout, non-static)**, which is 93.2%;
  - for **(nouns + adjectives, Thesaurus)** the best result is for **(augmented with $p = 0.25$, no dropout, non-static)**, which is 93.92%;
  - for **(nouns + adjectives, WordNet + Thesaurus)** the best result is for **(augmented with $p = 0.75$, no-dropout, non-static)**, which is 93.28%.
- There are two sets of variants for which DA always improves the baseline: **(nouns, WordNet + Thesaurus)** and **(adjectives, WordNet)**. There is one set of variants **(adjectives, Thesaurus)** that almost always improves the baseline (10 out of 12 times).

On the other hand, DA does not always improve the result. For some sets of variants, DA improves the baseline for each value of p; however, for other sets of variants, some values of p improve the baseline and others do not. This naturally brings up another research question: **which variants of data augmentation are better than others and why?**

The first finding is that the Thesaurus variants are generally better than the WordNet variants:

- The overall best result comes from Thesaurus variant: **(nouns + adjectives, augmented with $p = 0.25$, no dropout, non-static, Thesaurus)**.
- The variant **(nouns, Thesaurus)** is better than **(nouns, WordNet)** – the best result is 93.2% as compared to 92.76%. **(nouns, WordNet)** improves the baseline in only two cases, whereas **(nouns, Thesaurus)** improves it in five cases.
- The variant **(adjectives, Thesaurus)** the best result is 93.56%, whereas with **(adjectives, WordNet)**, the best is 93.48%. However, **(adjectives, Thesaurus)** improves the baseline 10 out of 12 times, whereas **(adjectives, WordNet)** always improves the baseline.
- The variant **(nouns + adjectives, WordNet)** improves the baseline seven times with a best result of 93.2%, whereas **(nouns + adjectives, Thesaurus)** also improves it seven times but yields the best result – 93.92% (the best overall).

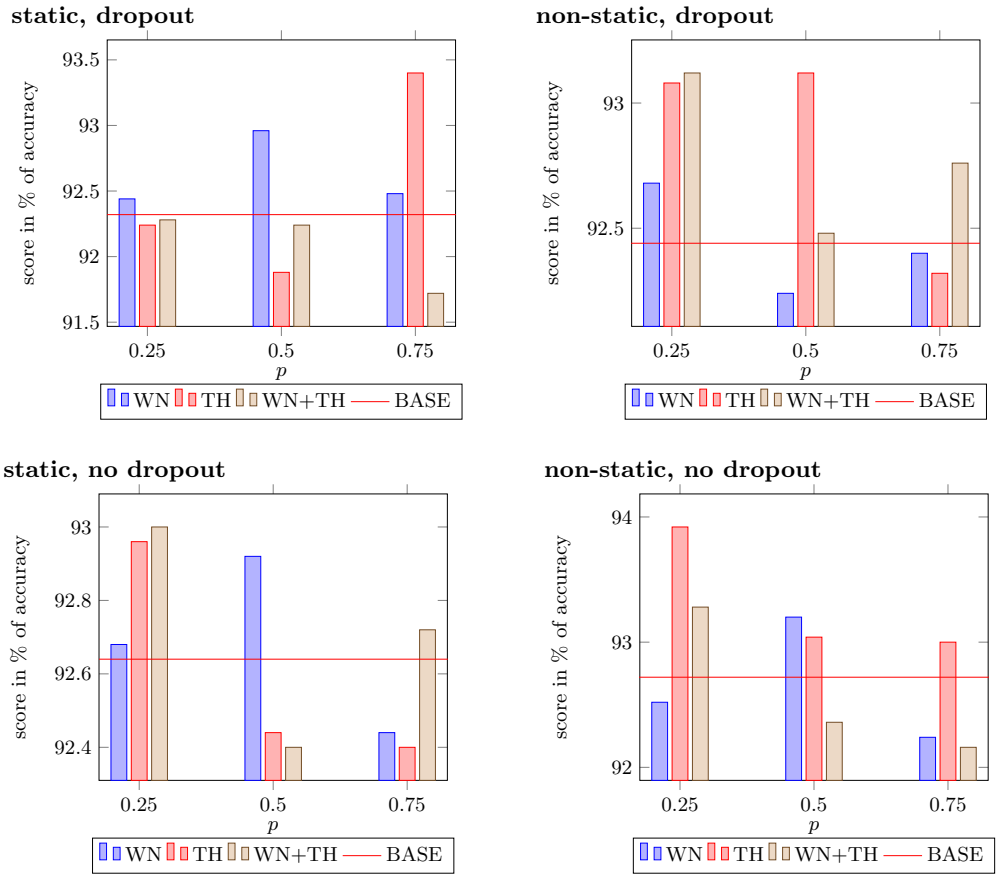

Figure 5. Results of classification on TREC dataset. Nouns and adjectives are replaced. Kim CNN static or non-static model with or without dropout. Scores for different values of p, which is probability of choosing transformed mini-batch for training instead of original mini-batch. WN means replacements are from WordNet, TH from Thesaurus, WN+TH – both. BASE is baseline result. Results present accuracy (AC); i.e., proportion of total number of predictions that were correct

The second finding is that broadening the source of synonyms may improve the best overall result; however, this also comes with a higher risk of the result dropping below the baseline:

- The variant **(adjectives, Thesaurus)** improves the baseline 10 out of 12 times with a best result of 93.56%, whereas **(nouns + adjectives, Thesaurus)** improves the baseline only 7 out of 12 times but yields a best result of 93.92% (which is the best overall).
- The variant **(adjectives, WordNet)** improves the baseline in all cases with a best result of 93.48%, whereas **(nouns + adjectives, WordNet)** improves the baseline 7 out of 12 times with a best result of 93.2%.

For this part of the experiments, we can say that the techniques for augmenting data we proposed generally improve the effectiveness of the Kim CNN model on the sentence-classification task. On the other hand, they do not improve it in all cases and variants. From our tests, it can be concluded that `Thesaurus.com` is generally a better source for replacing synonyms for DA than Princeton's WordNet. However, it is not entirely clear why this is the case. We provide another set of experiments with a different methodology and statistical significance analysis in the next section.

5.2. Further experiments and statistical significance tests

For this further analysis, we have used a different implementation of CNNs that is similar to the Kim CNN model; this was published by Sosuke Kobayashi on github [19, 20]. We conveyed the experiments without using dropout. As suggested by the author, further hyperparameters are shown in Table 10. This time, we used only Thesaurus as the source of synonyms. We tested on nouns, adjectives, or both.

Table 10
Suggested parameters of Kobayashi's implementation

Size of mini-batch	64
Optimization method	Adam [18]
The number of units of word embedding	256
Learning rate	0.001

We also adopted a different methodology for these tests, although we used the same TREC dataset. We ran the experiments for ten different seeds of a random number generator. For each seed, we compared the baseline CNN results with different variants of our DA method. We observed which of the DA variants had the best result on validation set and noted its result on the test set ("DA TEST for best VAL" column). We also show the best test result of all of the DA variants ("best DA TEST" column). We show the results in Table 11.

Table 11 shows that the result that performed the best on the validation set is better than the baseline in all cases. On the other hand, the difference was sometimes marginal; it ranged from 0.2% to 1.2%.

We also examined some of the cases for their statistical significance. We used the approach proposed by Thomas G. Dietterich in his work about statistical significance for supervised classifiers [10]. The author states that, among the different statistical tests, the one that is best-suited for comparing classification algorithms that are run once is McNemar's test.

McNemar's test consists of building a contingency matrix for two compared classifiers showing their relative performance on a test set. The matrix is 2×2, and its fields hold the number of examples classified correctly by both algorithms, labeled correctly by only one of them and incorrectly by both. We can tag these numbers as A, B, C, and D, respectively.

Table 11

Results of classification on TREC dataset with use of Kobayashi's implementation. The first column from left shows the number of experiment execution – each one for a different seed of random number generator. Next column shows the result of the baseline on the test dataset. Next three columns show the result on the test set of DA variant that performed best on the validation set, the name of variant, and the difference between this result and baseline. Finally, the next three columns show analogous information but for DA variant with the best result on the test set. Results present accuracy (AC); i.e., the proportion of the total number of predictions that were correct

No.	BASE [%]	Best VAL [%]	Variant	Diff [%]	Best [%]	Variant	Diff [%]
1	89.6	**90**	(both, $p = 0.25$)	**+0.4**	90.4	(adj, $p = 0.5$)	+0.8
2	89.8	**90.8**	(adj, $p = 0.75$)	**+1.0**	91	(adj, $p = 0.25$)	+1.2
3	89.8	**90.4**	(adj, $p = 0.25$)	**+0.6**	90.4	(adj, $p = 0.25$)	+0.6
4	90.2	**90.8**	(adj, $p = 0.25$)	**+0.6**	91.8	(both, $p = 0.25$)	+1.6
5	89.8	**91**	(adj, $p = 0.25$)	**+1.2**	91.8	(adj, $p = 0.5$)	+2.0
6	90.4	**90.6**	(adj, $p = 0.25$)	**+0.2**	90.8	(adj, $p = 0.75$)	+0.4
7	89	**90.2**	(adj, $p = 0.5$)	**+1.2**	90.8	(noun, $p = 0.75$)	+1.8
8	89.4	**91.4**	(adj, $p = 0.25$)	**+1.0**	92	(noun, $p = 0.5$)	+2.6
9	90	**90.2**	(adj, $p = 0.25$)	**+0.2**	91.2	(both, $p = 0.75$)	+1.2
10	90	**90.8**	(adj, $p = 0.5$)	**+0.8**	91	(adj, $p = 0.25$)	+1.0

We choose the best run for testing statistical significance; this was Run Number 5. The variant that performed the best on validation set (adj, $p = 0.25$) is 1.2% better than the baseline, and the best on test dataset (adj, $p = 0.5$) is 2% better. For the first case, we have A = 444, B = 5, C = 11, and D = 40. The p-value according to McNemar's test is 0.21. This indicates that this result is not significantly different from the baseline. In the second case (the best test result), we have A = 444, B = 5, C = 15, and D = 36. The p-value is 0.0414. This indicates that the best test result is significantly different from the baseline.

This shows that our proposed method has the potential for boosting a neural network's classification performance, although the improvement might rarely be statistically significant. Therefore, this paper shows an interesting basis for further research but does not propose a ready and ultimate technique for textual data augmentation.

6. Conclusions and future work

We presented a method for data augmentation used for text classification with convolutional neural networks. Our technique performed better than the baseline in most of the cases. The best of our variants performed 1.2% better than the baseline in terms of accuracy. We also conducted a partial statistical analysis of our results. We conclude that this work shows some promising results, although the proposed method is not a ready and ultimate technique for augmenting textual data with neural networks.

We leveraged a technique proposed for augmenting visual data [13]. It chooses data transformations that maximize the current loss of the classifier. We adapted it to the language domain, suggesting synonym replacement transformations using multiple language resources. As a CNN model, we used the one proposed by Yoon Kim [17] and another implemented by Sosuke Kobayashi [19, 20].

There are multiple possibilities for future work. One of them is examining other types of transformations, such as using the parse tree of a sentence instead of synonym replacement. Another idea is to test the DA method proposed by us in this work on other tasks and datasets. Eventually, all of these experiments might lead to the development of a more general DA method that could boost the performance of different neural network architectures on a variety of NLP tasks.

Acknowledgements

We used the computational resources of the Prometheus computer of the PLGrid infrastructure for the experiments described in this paper.

References

[1] Al-Rfou R., Choe D., Constant N., Guo M., Jones L.: Character-Level Language Modeling with Deeper Self-Attention, *CoRR*, vol. abs/1808.04444, 2018. `http://arxiv.org/abs/1808.04444`.

[2] app.dimensions.ai website. `https://app.dimensions.ai`.

[3] Bojanowski P., Grave E., Joulin A., Mikolov T.: Enriching Word Vectors with Subword Information, *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[4] Bottou L.: Large-scale machine learning with stochastic gradient descent. In: *Proceedings of COMPSTAT'2010*, pp. 177–186. Springer, 2010.

[5] Cireşan D., Meier U., Masci J., Gambardella L.M., Schmidhuber J.: Flexible, high performance convolutional neural networks for image classification. In: *Twenty--Second International Joint Conference on Artificial Intelligence*, pp. 1237–1242, 2011.

[6] Cireşan D., Meier U., Schmidhuber J.: Multi-column deep neural networks for image classification. In: *IEEE conference on Computer vision and pattern recognition (CVPR)*, pp. 3642–3649, 2012.

[7] Collobert R., Weston J., Bottou L., Karlen M., Kavukcuoglu K., Kuksa P.: Natural Language Processing (almost) from Scratch, *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[8] Coulombe C.: Text Data Augmentation Made Simple By Leveraging NLP Cloud APIs, *CoRR*, vol. abs/1812.04718, 2018. `http://arxiv.org/abs/1812.04718`

[9] Devlin J., Chang M.W., Lee K., Toutanova K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, *CoRR*, vol. abs/1810.04805, 2018. http://arxiv.org/abs/1810.04805.

[10] Dietterich T.G.: Approximate statistical tests for comparing supervised classification learning algorithms, *Neural Computation*, vol. 10(7), pp. 1895–1923, 1998.

[11] Dodge J., Gane A., Zhang X., Bordes A., Chopra S., Miller A., Szlam A., Weston J.: Evaluating Prerequisite Qualities for Learning End-to-End Dialog Systems, *CoRR*, 2015. https://arxiv.org/abs/1511.06931.

[12] Faucett L.: *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1994.

[13] Fawzi A., Samulowitz H., Turaga D., Frossard P.: Adaptive data augmentation for image classification. In: *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 3688–3692, 2016.

[14] Gehring J., Auli M., Grangier D., Yarats D., Dauphin Y.N.: Convolutional sequence to sequence learning. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1243–1252, 2017.

[15] Goodfellow I., Bengio Y., Courville A.: *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[16] Harvard NLP, Kim CNN implementation. https://github.com/harvardnlp/sent-conv-torch.

[17] Kim Y.: Convolutional Neural Networks for Sentence Classification. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746–1751. Association for Computational Linguistics, 2014. http://dx.doi.org/10.3115/v1/D14-1181.

[18] Kingma D.P., Ba J.: Adam: A Method for Stochastic Optimization, *CoRR*, vol. abs/1412.6980, 2015. https://arxiv.org/abs/1412.6980.

[19] Kobayashi S.: Contextual Augmentation: Data Augmentation by Words with Paradigmatic Relations. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 452–457. Association for Computational Linguistics, 2018. http://dx.doi.org/10.18653/v1/N18-2072.

[20] Kobayashi S.: CNN implementation. https://github.com/pfnet-research/contextual_augmentation.

[21] Krizhevsky A., Sutskever I., Hinton G.E.: ImageNet classification with deep convolutional neural networks. In: *Advances in neural information processing systems 25 (NIPS 2012)*, pp. 1097–1105, 2012.

[22] LeCun Y.: Une procedure d'apprentissage pour reseau a seuil asymetrique. In: *Proceedings of Cognitiva 85*, pp. 599–604, 1985.

[23] LeCun Y., Bengio Y., Hinton G.: Deep learning, *Nature*, vol. 521(7553), p. 436, 2015.

[24] Li X., Roth D.: Learning question classifiers. In: *Proceedings of the 19th international conference on Computational linguistics – Volume 1*, Association for Computational Linguistics, pp. 1–7, 2002.

[25] Lowe R., Pow N., Serban I., Pineau J.: The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. In: *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Association for Computational Linguistics, pp. 285–294, 2015. http://dx.doi.org/10.18653/v1/W15-4640.

[26] Manning C.D.: Computational linguistics and deep learning, *Computational Linguistics*, vol. 41(4), pp. 701–707, 2015.

[27] Mikolov T., Chen K., Corrado G., Dean J.: Efficient Estimation of Word Representations in Vector Space, *CoRR*, vol. abs/1301.3781, 2013. http://arxiv.org/abs/1301.3781.

[28] Mikolov T., Sutskever I., Chen K., Corrado G.S., Dean J.: Distributed Representations of Words and Phrases and their Compositionality. In: *Advances in neural information processing systems 26 (NIPS 2013)*, pp. 3111–3119, 2013.

[29] Miller G.A.: *WordNet: An electronic lexical database*. MIT Press, 1998.

[30] Miller G.A.: WordNet: a lexical database for English, *Communications of the ACM*, vol. 38(11), pp. 39–41, 1995.

[31] Parker D.B.: *Learning Logic Technical Report TR-47*,Center of Computational Research in Economics and Management Science, Massachusetts Institute of Technology, Cambridge, 1985.

[32] Paulin M., Revaud J., Harchaoui Z., Perronnin F., Schmid C.: Transformation pursuit for image classification. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3646–3653, 2014.

[33] Pennington J., Socher R., Manning C.: GloVe: Global Vectors for Word Representation. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[34] Ptaszyński M., Leliwa G., Piech M., Smywiński-Pohl A.: Cyberbullying Detection – Technical Report 2/2018, Department of Computer Science AGH, University of Science and Technology, *CoRR*, vol. abs/1808.00926, 2018. http://arxiv.org/abs/1808.00926.

[35] PyDictionary. http://pypi.org/project/PyDictionary/.

[36] Quijas J.K.: *Analysing the effects of data augmentation and free parameters for text classification with recurrent convolutional neural networks*, Master Thesis, The University of Texas at El Paso, 2017.

[37] Ratner A.J., Ehrenberg H., Hussain Z., Dunnmon J., Ré C.: Learning to Compose Domain-Specific Transformations for Data Augmentation. In: *Advances in Neural Information Processing Systems*, pp. 3239–3249, 2017.

[38] Rosario R.R.: *A Data Augmentation Approach to Short Text Classification*, Ph.D. thesis, University of California, Los Angeles, 2017.

[39] Rumelhart D.E., Hinton G.E., Williams R.J.: Learning representations by back--propagating errors, *Nature*, vol. 323(6088), pp. 533–536, 1986.

[40] Simard P.Y., Steinkraus D., Platt J.C.: Best practices for convolutional neural networks applied to visual document analysis. In: *Proceedings of Seventh International Conference on Document Analysis and Recognition, 2003, Edinburgh, UK*, vol. 3, pp. 958–962, 2003.

[41] Socher R., Lin C.C., Ng A.Y., Manning C.: Parsing natural scenes and natural language with recursive neural networks. In: *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 129–136, 2011.

[42] Srivastava N., Hinton G.E., Krizhevsky A., Sutskever I., Salakhutdinov R.: Dropout: a simple way to prevent neural networks from overfitting, *Journal of Machine Learning Research*, vol. 15(1), pp. 1929–1958, 2014.

[43] Thesaurus.com. `www.thesaurus.com`.

[44] Toutanova K., Klein D., Manning C.D., Singer Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology – Volume 1*, pp. 173–180, Association for Computational Linguistics, 2003.

[45] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A.N., Kaiser Ł., Polosukhin I.: Attention is all you need. In: *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pp. 5998–6008, 2017.

[46] Werbos P.: *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*, PhD thesis, Harvard University, 1974.

[47] Wong S.C., Gatt A., Stamatescu V., McDonnell M.D.: Understanding data augmentation for classification: when to warp? In: *Digital Image Computing: Techniques and Applications (DICTA), 2016 International Conference on*, pp. 1–6, 2016.

[48] WordNet online. `wordnet.princeton.edu`.

[49] Wu Y., Schuster M., Chen Z., Le Q.V., Norouzi M., Macherey W., Krikun M., Cao Y., Gao Q., Macherey K., Klingner J., Shah A., Johnson M., Liu X., Kaiser Ł., Gouws S., Kato Y., Kudo T., Kazawa H., Stevens K., Kurian G., Patil N., Wang W., Young C., Smith J., Riesa J., Rudnick A., Vinyals O., Corrado G., Hughes M., Dean J.: Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation, *CoRR*, vol. abs/1609.08144, 2016. `http://arxiv.org/abs/1609.08144`.

[50] Young T., Hazarika D., Poria S., Cambria E.: Recent trends in deep learning based natural language processing, *IEEE Computational intelligence magazine*, vol. 13(3), pp. 55–75, 2018.

[51] Zeiler M.D.: ADADELTA: an adaptive learning rate method. In: *arXiv preprint arXiv:1212.5701*, 2012.

[52] Zhang X., Zhao J., LeCun Y.: Character-level convolutional networks for text classification. In: *Advances in neural information processing systems*, pp. 649–657, 2015.

[53] Zhou X., Dong D., Wu H., Zhao S., Yu D., Tian H., Liu X., Yan R.: Multi-view response selection for human-computer conversation. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 372–381, 2016.

Affiliations

Michał Jungiewicz
AGH University of Science and Technology, Faculty of Computer Science, Electronics and Telecommunications, Department of Computer Science, Krakow, Poland, mjungiew@agh.edu.pl

Aleksander Smywiński-Pohl
AGH University of Science and Technology, Faculty of Computer Science, Electronics and Telecommunications, Department of Computer Science, Krakow, Poland, apohllo@o2.pl, ORCID ID: https://orcid.org/0000-0001-6684-0748

Received: 21.08.2018
Revised: 04.03.2019
Accepted: 04.03.2019